Faculty of Engineering and Information Technology
Computer Science Department
COMP2311 Project (Phase 2)

Project title: Grocery Store Inventory

Project objective: 1) To create GUI using JavaFx framework
2) To apply the concept of Event-Driven Programming

Project Description:

As your first job fresh out of college, you are hired on as a bagger at the local supermarket. To impress the store manager, you decide to write an inventory program to help with inventory tracking. Because you have your heart set on moving into the exciting and fast-paced world of grocery software, you hope that your Inventory program will be a prototype for many future grocery software solutions. **This has been done at the first phase of your project.**

As a result of the first phase of your project, your manager became happy with your contribution. To encourage you, the manager asked you to keep improving your software so that he can raise your salary. Due to the manager's age, he needs user-friendly and easy-to-use software for his grocery software.

As part of this phase of your project, we ask you to add new rich JavaFX components. There is no need to change the basic functionalities you implemented in the first phase, but you must convert all the statements you tested in the driver class into GUI components. Output should be displayed on the user interface and files instead of the console.

Details

#Note: eclipse is strongly encouraged as the Integrated Development Environment (IDE) for this assessment. You will **receive a zero mark for using a different IDE.**

#Note: Create a JavaFX application with a graphical user interface (GUI) based on the different figures in this file (Swing and AWT are not part of JavaFX and are not acceptable for this application).

Write code to display each of the following screens in the GUI:

A. <u>A main screen, showing the following controls. (Figure 1)</u>:

- A label with the text "Inventory Management System: Comp 2311 Project, Phase 2"
- The image shows that this software is related to Inventory System. The project files include a copy of the image.
- The buttons "Add", "Modify", "Delete", and "Stock Report" open new windows that offer different features.
- The application title is "Inventory Management System".

Figure 1: Inventory Management System main window

B. <u>The add button screen shows the following controls</u> (Figures 2 & 3):

- For brands, there are radio buttons that say "Yes" and "No"
- There are buttons for "Save" and "Cancel"
- There are labels for "Type", "Quantity", and "Price".
- There are text fields for "Type", "Quantity", and "price".
- Label to display messages like the ones you did in the first phase.
    - For example: **(bread already exists)**
      (bread - in stock: 15, price: $9.99)

see the output from the first phase of the project

- There is a title for the window "Add New Item"

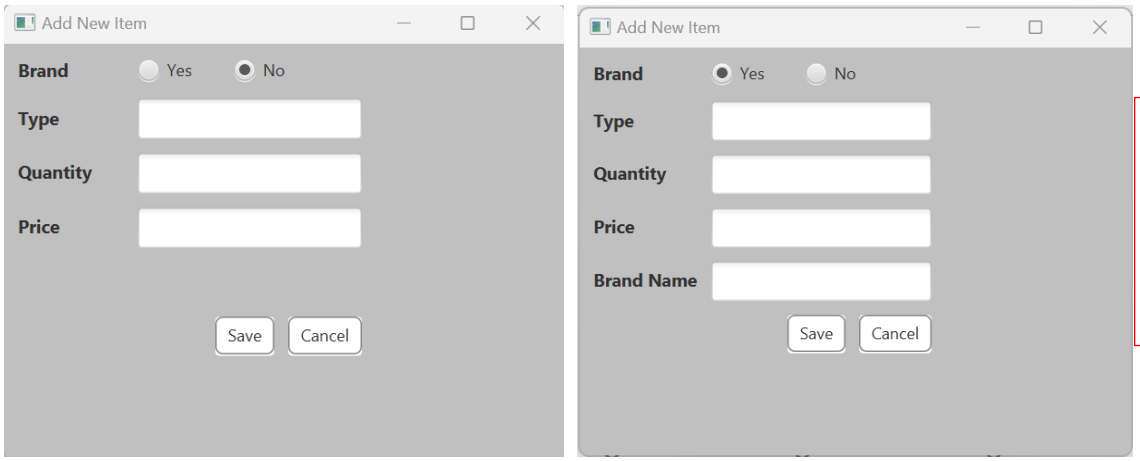
For clarification:

Clicking "**Yes**" will display a **label called "Brand name" and a text field for "Brand name"**. These controls will **disappear if the user clicks "No".** See Figure 2.

It is important to remember that some items from the first phase have a brand name and some do not.

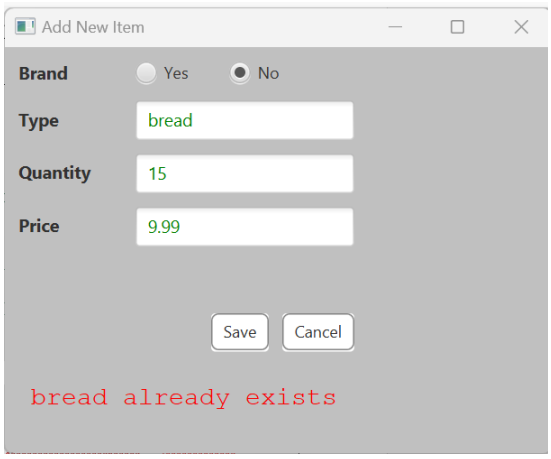
Clicking a "Save" button, will do the job of the newItem method (check the first phase of the project).

***Cancel Button*: The implementation behind this button is up to you (Don't leave it empty).**



Consider the difference between selecting "No" and selecting "Yes."

Figure 2: Add New Item window



Sample Run:
Adding a "bread" and clicking save. See the message "bread already exists"

Figure 3: Sample run

C. The modify button screen shows the following controls (Figures 5, 6 & 7):

- For brands, there is a ComboBox with the options "Brand" and "Not Brand."
- There are "Update" and "Cancel" buttons.
- There are labels for "Type," "Quantity," and "Price".
- There are text fields for "Type," "Quantity," and "Price".
- ***There is "label" to display messages similar to those displayed in the first phase***.
- The window's title is "Update Item."

For clarification:

Selecting "Not Brand" from the ComboBox will display a "Search" button (figure 5), while selecting "Brand" will display a text field for "Brand name" in addition to the "Search" button (Figure 6). You can make the ComboBox editable, which means you can type "brand" or "not brand" without selecting anything (Figure 4).

At first, none of the text fields are editable (It is not possible to enter anything in the text fields until you have searched and gotten a result).

Once you click the "Search" button, the details of the item you want to update will appear. In case the item already exists, you can update the fields, otherwise, an error message will appear (Figure 7).

A click on the "Update" button will carry out the same function as the update method described in the first phase.

The user should see a successful message if the item successfully updates; otherwise, an error message will appear.

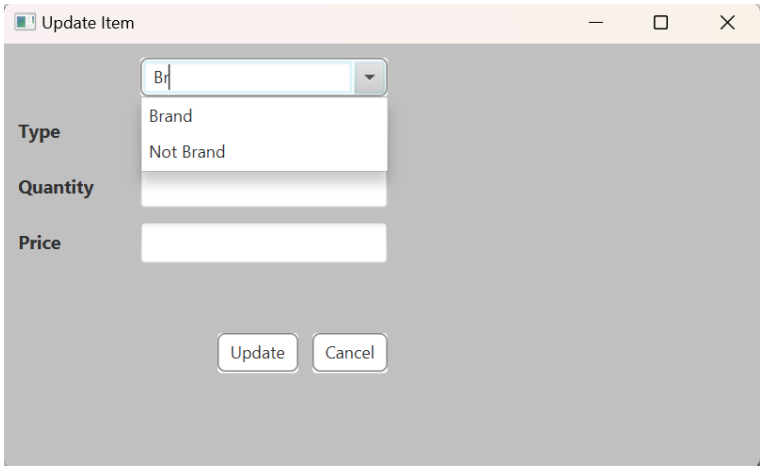


Figure 4: Update Item Window, editable ComboBox

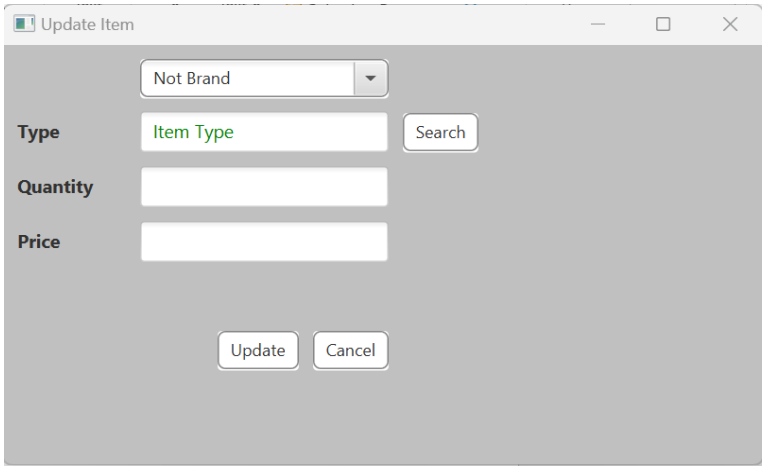


Figure 5: Sample run, selecting “Not Brand” from the ComboBox

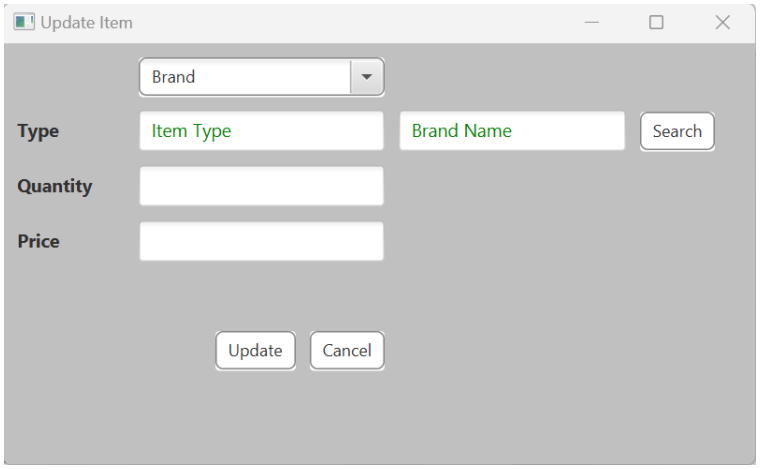


Figure 6: Sample run, select “Brand” from the ComboBox.

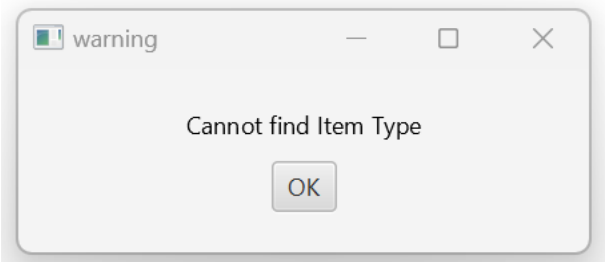


Figure 7: Warning message

D. The delete button screen shows the following controls (Figures 8):

- There is a ComboBox for brands that shows two options: "Brand" and "Not Brand"
- There are "Delete" and "Cancel" buttons
- There are labels for "Type", "Quantity", and "Price".
- There are text fields for "Type", "Quantity", and "price".
- The "label" displays messages indicating deletion status (error, success, etc.).
- "Delete Item" is the title of the window.

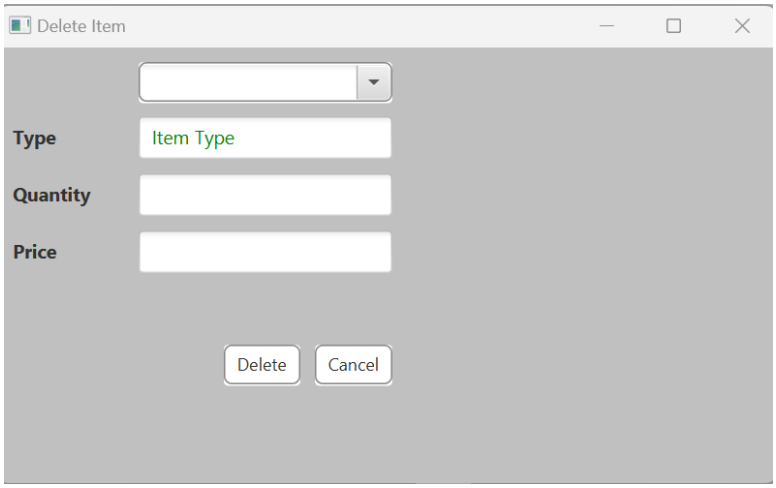


Figure 8: Sample run

For clarification:

- Like the Update window, the Delete window functions similarly, except that the item must be deleted instead of its details being updated.

- Phase 1 does not have a method for removing items from inventory. You must implement this method, as it is considered a new method at this phase.

E. The stock report button screen shows the following controls (Figures 8):

- There is a Text component that displays "The following options can be used to print a report"
- There are two check boxes for printing the report: "Export a copy to a file" and "TextArea".
- There is a TextField for "fileName".
- There is a button for “Export”.
- There is a TextArea.
- "Stock Report" is the title of the window.

For clarification:

The stock report includes the same information that exists in phase one of the project. The only difference is the way of printing. Instead of printing on the console. Here you have three options for printing depending on the choices of the check box:

1. Export the stock report to a text file by selecting “Export a copy to a file”, write the name of the file, and then click export.
2. Display the information in the text area by selecting the “TextArea option” .
3. Display the stock report in both the “text area” and in the “text file” by selecting both options “Export a copy to a file” and “TextArea” option.

In the beginning, no options are selected, see Figure 9. After selecting, the “Export a copy to a file”, a text field, and an “Export” button will show, see Figure 10. By Selecting “TextArea”, a Text Area will be appeared, see Figure 11. Finally, selecting both options will leads to appear all the controls, see Figure 12.

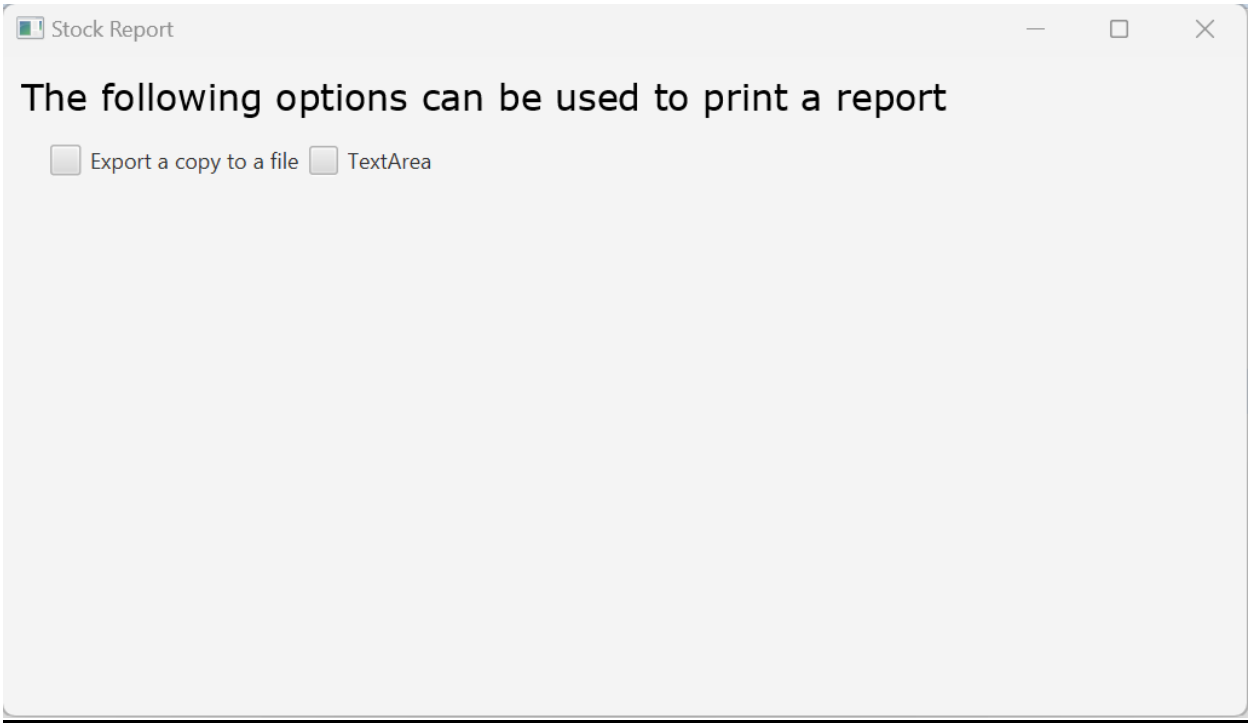


Figure 9: Sample run: no options are selected

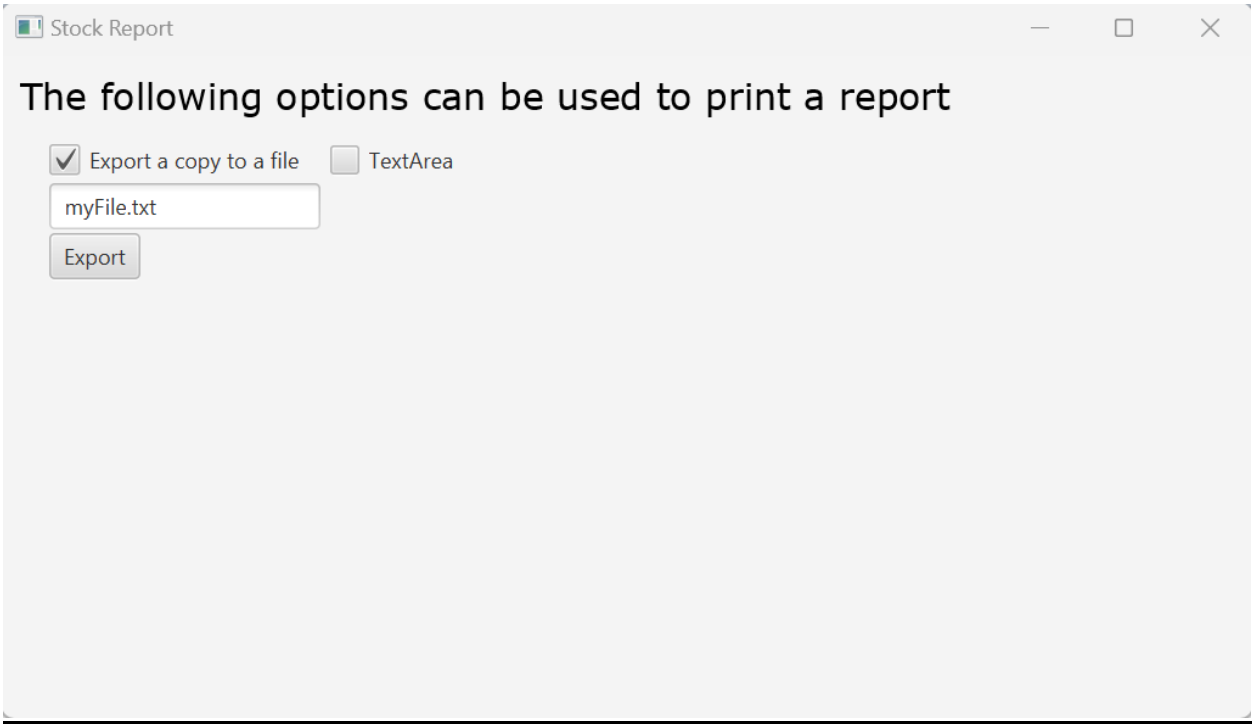


Figure 10: Sample run: selecting “Export a copy to a file”

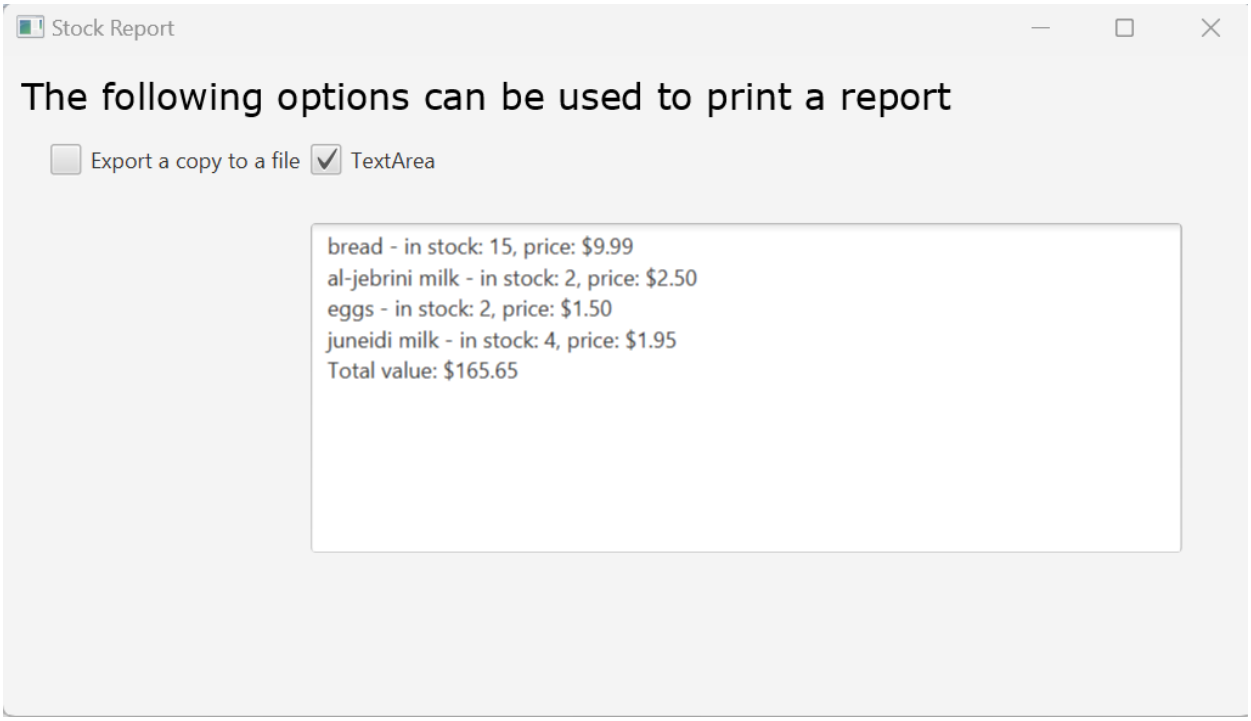


Figure 11: Sample run: selecting a “TextArea”

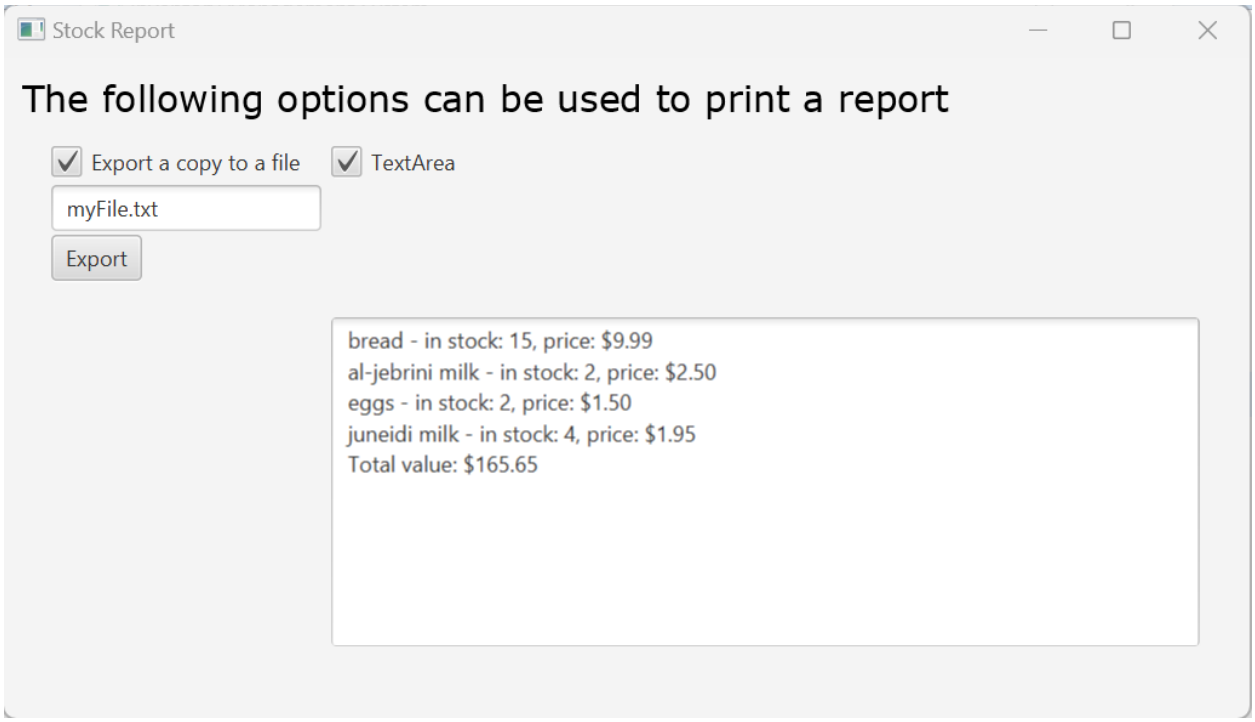


Figure 12: Sample run: selecting both options

Very important notes to consider (not considering them leads to losing points):

<Inheritance Relation>

- In this project, it is mandatory to take advantage of the inheritance relationship, so maybe one or two stages will need to be built and then the others can be extended and customized as needed.

<CSS for the interface>

- You must use the "rotation and the set style methods" that exist in the node to have something like the illustrated figures. Setting colors or backgrounds without using the set style method will lead to loss of some marks.

<Exception handling>

- You must handle all the "checked" and "unchecked" exceptions (e.g.: NullPointerException, ArithmeticException, FileNotFoundException, etc.)

<Developed methods from the first phase>

- It is intended that all the methods developed in the first phase will be utilized in the second phase as well.

<Layout>

- You can use any layout that you feel matches the illustrations (the figures above).

<Event handling>

- The use of "lambdas" or "anonymous inner classes" to simplify event handling is not allowed. **You must write a complete handler class** that includes all the details. However, **multiple events can be handled with a single handler class**.

<Read from file>

- As part of your program, you must read items (with and without brands) from a file called "inventoryDatabase.txt" and store them in the Inventory ArrayList.

  You have two options to do that (do any one of them):
  - Your program will automatically load the items from the file when the main window appears.
  - The main window should include a "FileChooser" that is used to browse the files from the system, select the file and read it.

Please note the Followings:

1. This is an **individual work project**. There will be disciplinary action taken against those who **cheat**.
2. Based on Java formal documentation, your program should be **well commented**.
3. **Submissions**:
   3.1 Due date: **Tuesday, January 24, 2022, at 11:59 pm.**
   3.2 Make sure you submit your project through **ITC before the deadline**.
   3.3 Your project should be submitted as a **compressed file** in the following **format**: (**yourId_yourName_Section#**), for example, **1020077_Sandy Ahmad_Section 4.rar**.

"Success is the sum of small efforts, repeated day in and day out."

Robert Collier

Best of Luck